Presented at 1998 German Workshop on Case-Based Reasoning

The Limits of CBR in Software Project Estimation

Sarah Jane Delany[1], Pádraig Cunningham[2] and Wolfgang Wilke[3]
[1]Dublin Institute of Technology, Ireland
[2]Trinity College Dublin, Ireland
[3]University of Kaiserslautern, Germany.

Abstract. Software project cost estimation is difficult because of problems of quantifying project size and because of the continual emergence of new technology. This presents as a classic example of a weak theory domain where experience is key and appears well suited to CBR. Indeed there are several reports in the literature on the use of CBR in project cost estimation. Research described to date has focused on applications late in the development life cycle and in a narrow domain. In this paper we describe existing research and explore the use of CBR beyond these limits. We look at the use of CBR for estimation earlier in the life cycle and for use in broader domains where more abstract remindings need to be supported.

1. Introduction

Case-based reasoning (CBR) may function as a corporate memory where experience is stored as cases and may be drawn upon when new problems arise. The effectiveness of CBR at an operational level has been proven with many successful applications such as help desks and credit risk assessment systems in operation. Indeed there is a lot of research on the use of CBR at an operational level in software project cost estimation which is our interest here. In this paper we explore the potential for CBR in project cost estimation at a more strategic level in the process and across broader domains.

Software project cost estimation is problematic because it is difficult to derive accurate size and cost figures from the features of a project that are known early in the development process. This is characteristic of *weak theory domains* where CBR has potential as a solution. Existing CBR approaches to software project cost estimation use conventional estimation techniques such as COCOMO or Function Point Analysis to build a case structure (see Section 2 for details). This research is more successful that conventional techniques because CBR avoids the need to model how these estimates of size contribute to the actual cost. Research reported so far is limited in two ways. It is focused late in the development cycle when an estimate of size is already available (e.g. function point count). It has been applied in narrow domains; there has been little attempt to perform estimation across different development platforms. In this paper we discuss what is required to go beyond these limits. In section 2 software project estimation is reviewed. In section 3 and 4 the requirements of earlier estimation and estimation across broader

domains are discussed. The paper concludes with some discussion and directions for future work.

2. Software Project Estimation

Software is expensive to develop and it is a major cost factor in corporate information systems' budgets. With the variability of software characteristics and the continual emergence of new technologies it is becoming harder and harder to correctly estimate software development costs. With products developed for the mass markets the cost of development is not visible in the price of the product. However, with custom or bespoke development, the software is targeted for one or a small number of customers and the development cost influences the price. It is of strategic importance for an organisation, whether as a customer or a developer, to be able to base its purchase or sales decisions on the ability to estimate the cost of development correctly and consistently.

Estimation involves the identification and quantification of the factors involved in the development of a software product. These include the processes by which the software is developed, the personnel and support resources used in the development and the nature of the product itself. Predicting the cost and duration of a project still remains a problem. Jack and Mannion (1995) conclude from surveys of organisations that only 25% of projects come within their originally predicted cost and schedule, that 66% of companies significantly underestimate the time and cost and that the costs for similar projects can vary by up to 200%.

There are a number of different estimation techniques currently in use in industry and these can be arranged into 3 main categories:

- Algorithmic models which predict effort and duration as a function of a number of variables. The most common algorithmic models in use are COCOMO (COnstructive COst MOdel) (Boehm 1981) and Function Point Analysis (Albrecht & Gaffney 1983).
- Expert judgement involving predictions based on the skill and experience of one or more experts.
- Estimation by Analogy involving the comparison of one or more completed projects to a similar new project to predict cost and schedule.

There have been a number of studies attempting to evaluate the effectiveness of algorithmic cost models. Bredero et al. (1989) summaries the results of 17 different studies into cost estimation and software sizing models and tools. This research into the use of the algorithmic models for cost estimation has shown that the models perform badly. Actual values of effort usually differ significantly from the predicted values, even when all the input variables are known. Also the results of estimated effort or duration derived from different models are very different. Studies such as those undertaken by Kemerer (1987) and Abdel-Hamid & Madnick

(1987) which have attempted to show that selected models are accurate and can be used for estimating projects, have not been able to prove their hypotheses.
In addition, there is significant evidence that calibration of an algorithmic model with an organisation's historic project data is crucial and research has suggested that local models are more accurate that general purpose models (Kemerer (1987); Jack & Mannion 1995; Cuelanaere *et al*. 1987).
There has not been much research to examine the use of expert judgement to estimate development effort or duration. However, according to Wrigley and Dexter (1987), expert judgement is still the most dominant method of estimation. Vicinanza *et al.* (1991) concluded that experienced managers can make more accurate estimates that existing (uncalibrated) algorithmic models, specifically COCOMO or Function Point Analysis.

2.1 Project Estimation Using CBR

More recently research has been undertaken in the area of estimation by analogy and more specifically the application of CBR to cost estimation. CBR (Kolodner 1993, Watson & Marir 1994; Barletta 1991) involves matching the current problem against similar problems that have occurred in the past. It is attractive as a technique to apply to cost estimation as it uses past experiences to solve new problems which corresponds to how experts operate.
Recent research has shown the feasibility of applying CBR to the problem of project cost estimation (Finnie *et al.* 1997; Shepperd *et al.* 1996; Bisio & Malabocchia 1995; Mukopadhyay *et al.* 1992) and a number of CBR applications applied to the problem of software effort estimation have been developed. (Prietula *et al.* 1996; Bisio & Malabocchia 1995; Shepperd *et al*. 1996).

2.2 Focus of existing work

The current research into applying CBR to cost estimation focuses on problem domains that are similar in nature - similar types of applications from similar organisations. The data sets used by Prietula *et al.* (1996) represented "a typical mix of data processing projects (mostly COBOL)". Shepperd & Schofield (1996) used data sets from a variety of sources including large business applications, enhancements to telecommunications software and IBM data processing services projects. A significant portion of the data sets are dated between 1983 and 1988 which would not take into account the variety of platforms and new technologies that contribute to software development in the 1990s.
In addition, examination of the features or attributes that contribute to the case representation used in the current research imply that a detailed specification of software was available at the time that the software cost estimation was undertaken. Prietula *et al.* (1996) includes case factors such as "number and

complexity of files generated, used and maintained" and "number and complexity of reports generated", while in their case study Shepperd & Scholfield (1996) use "number of parameters used by the subscriber input procedures" and "number of messages passed between blocks". Finnie *et al.* (1997), while using research data from 299 projects from 17 different organisations used attributes such as numbers of inputs, outputs, internal and external files to identify similar cases.
These factors lead to the current research being placed in the context of similar applications and applying late in the software development life cycle, as represented by Figure 1.

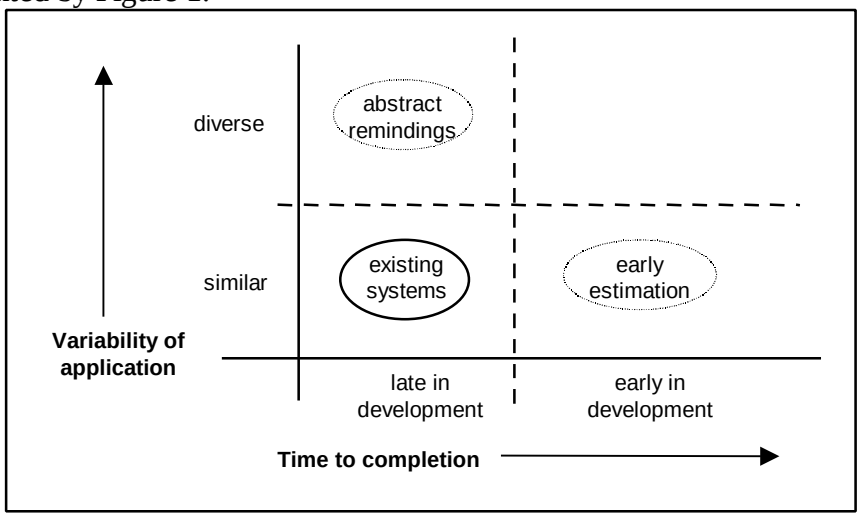

Fig. 1: Positioning of current research

3. Early Estimation

The accuracy of software estimates has a direct impact on the quality of an organisation's software investment decisions. Accurate estimation as early as possible in the development life cycle is important as the initial estimates will input to the cost/benefit analysis and influence decisions as to whether projects are strategically viable.
In order to estimate development effort at an early stage in the development life cycle it is necessary to identify the attributes of the project that are predictive of the effort involved. One of the issues with estimation at an early stage in the life cycle is the availability of enough predictive features that can give an accurate estimate of effort. Most of the algorithmic models use a measure of system size as a key input to the estimation procedure. COCOMO uses lines of code as a measure of software size while Function Point Analysis uses a measure of system size called function points which is derived from design features such as the number of inputs, outputs and entities in the software system. The existing CBR applications for cost estimation referenced above also use design features and metrics which cannot be

known with any reasonable certainty early in the life cycle at the time of initial cost estimation.
To be successful in applying CBR to early cost estimation will involve identifying more abstract features than metrics such as size or specific detailed design features to represent a case. These features must still contribute to determining the effort involved in software development. With current advances in technology and the trends towards system integration, it is often the environmental factors such as stability of user requirements, the commitment of a project sponsor, maturity of the technology etc. that contribute to the estimates. Consequently, these environmental factors contribute to the success or failure of a software development project. However, features such as these may not lead to an estimate which is an actual measure of effort involved in the design and development of the software. These types of features may better be used to estimate a measure of the uncertainty associated with the project due to the effect of the environmental factors.

3.1 CBR in early estimation

One simple way of calculating the effort involved in the development of a software system requires a measure of size of the system (in lines of code or function points, for example) multiplied by the expected productivity of the project team (such as the number of lines of code or function points per day).

$$\text{Effort} \propto \text{Size} * \text{Productivity}$$

One of the most difficult issues to deal with in estimating effort in early estimation is providing a measure of the system size. Shepperd *et al.* (1996) describe using analogy for effort estimation and state that it is important to choose at least one variable or feature to act as a size driver. At early stages in the development life cycle this size driver which is used to produce the estimate, is itself an estimate. Without some design work it is impossible to accurately estimate the number of inputs, screens or classes, for example. For this reason it is not reasonable to expect an accurate estimate of effort (e.g. in man months). It is more appropriate to use the available predictive features to come up with a weighing which will indicate the effect that the case features will have on the expected productivity of the project. Let us call this weight the productivity coefficient.
Average productivity across organisations can be very different. Existing research on cost estimation techniques has demonstrated that local calibration within organisations is necessary (Kemerer 1987; Jack & Mannion 1995; Cuelanaere *et al*. 1987). The collection of data from previously completed local projects is necessary for successful estimation (Heemstra 1992). A productivity coefficient for cases in the case-base could be calculated as the ratio of the actual effort (available after the project is completed) to the average development productivity of the organisation

calibrated across all the cases in the case-base. This coefficient is not an actual measure of risk but is an indication of an organisation's potential productivity based on its past experiences with 'similar' development projects.

3.2 Cost Drivers

Cost drivers are the variables that are believed to influence the cost of software development. A study by Noth and Kretzschmar (1984) found that more than 1200 cost drivers were mentioned in the literature. Most of the algorithmic models use cost drivers in their calculation of the estimate - COCOMO uses 15 cost drivers which include such variables as analyst capability, use of software tools and computer turnaround time.

To identify a case representation for a software development project we attempt to identify those cost drivers that have the most influence on the development effort of a software system. These cost drivers need to be available early in the development stage of a project - at the project specification stage.

The main issues with cost drivers are that there is a lack of clear definitions for variables, such as quality, complexity and experience. Also the majority of cost drivers are difficult to quantify and can be very subjective. Due to this and the fact that local calibration is important this CBR solution will only be applicable locally within organisations. Used across organisations may result in inconsistent definitions of variables or quantification and subjectivity in the rating of the features.

3.3 Case representation

Figure 2 presents the proposed case representation. A number of the features in the case representation refer to the influence and experience of the management of a software development project. "Management can have more influence on the productivity of the programming staff than any technology now in use" (Kendall & Lamb 1977). The variables are identified as risk variables or variables contributing to project overruns or causing inaccurate estimates in various studies (Barki *et al.* 1993; Subramanian & Breslawski 1994; Subramanian & Breslawski 1995; Heemstra 1992). Subramanian & Breslawski (1995) also found that using a project manager's experience to adjust the estimates derived from algorithmic models leads to improved accuracy in effort estimation.

The composition and experience of the project team is also key in effecting the development estimates (Heemstra 1992; Barki *et al.* 1993; Boehm & Papaccio 1990; Jeffery 1987). Heemstra (1992) concludes that the 'human aspects' are very important in software cost estimation. Features such as quality, experience and composition of the project team, the degree to which the project manager can

motivate and encourage his team will have more influence in delivering a project within time and budget than use of any models.

Feature
Management Top management support/commitment Project manager's experience (may be number of projects) Project manager's success rating (some indication of performance on previous projects) Manager's familiarity with team Project Team Team IT experience Team understanding/experience of application Users User understanding of requirements Extent of user support User IT competence and experience Application Requirements stability Required reliability Method Technology (can be expanded to different types, e.g. object, client/server, distributed, etc appropriate to an organisation) Use of productivity tools Use of standards
Productivity Coefficient Actual Effort Actual Size

Fig. 2: Case representation

Research has also identified the importance of significant user participation and experience in successful projects (Barki *et al* 1993; Lederer & Prasad 1992; Subramanian & Breslawski 1995).

A number of the cost drivers in use in current estimating techniques regarding the application itself are variables that are dependent on a certain amount of design being completed, such as software complexity or database size. Those application variables that influence the estimate and are available at an early stage in development are an indication of the likely stability of the requirements, the reliability required of the application and the required integration with other systems.

Lastly considering the method of development, studies have shown that use of productivity tools, certain technologies and standard design techniques can

influence the estimate (Lederer & Prasad 1992; Subramanian & Breslawski 1994; Heemstra 1992). Lederer & Prasad (1992) also concluded that the accuracy of estimates improves with the use of documented facts and standards.

4. Estimation Across Broader Domains

In order to use CBR in estimation in a broader context there is need to identify abstract features that will support remindings across different contexts and a need to develop adaptation techniques to transform cases between these contexts.

The use of concrete software metrics and concrete design features does not seem feasible for the prediction of software development cost between diverse applications. There is the need to find abstract characteristics which are invariant in the transformation to completely different applications. For example COCOMO (Boehm 1981) uses a set of personnel attributes to determine the influence between staff capabilities/experience and the resulting development effort. Further, project attributes are used to describe the influence of the use of software tools, like debuggers or GUI builders, on the software development costs. These are examples of software project characteristics which might be usable across different types of applications. Nevertheless there is the need to extend current known characteristics and to evaluate their usability across diverse applications.

CBR also requires the facility to adapt old solutions to solve new problems. These adaptation techniques can be used to build a bridge between experiences in diverse applications. In experiments with data provided by the Australian Software Metrics - Association (ASMA), Finnie & Wittig (1996) uses adaptation rules to transform experiences across diverse projects. They adapt old cost estimation experience across different classes of programming languages (from 3GL to 4GL and vice versa) or across different hardware platforms such as PCs, midrange and mainframes. They use measures for the average productivity difference between 3GL and 4GL or the productivity difference between the different platforms. This adaptation step addresses the transformation of cost estimations across diverse applications.

It may be the case that in such an ‘open’ problem as cost estimation across domains it would be prudent to pursue interactive adaptation rather than fully automated adaptation. The best use of CBR might be to produce useful reminders for the expert or perhaps good estimates for input parameters into a parametric model.

5. Conclusions & Future Work

Current research on using CBR for software project cost estimation has focused on estimation late in the development cycle and within narrow domains. The two directions in which this can be extended are to support earlier estimation and

estimation across broader domains. Both of these directions require the identification of more abstract features to support the more complex remindings needed.

For early estimation, an accurate measure of size is not available early in the development life cycle so we advocate that it would be pragmatic to shift the objective from cost estimation to the estimation of a productivity coefficient that describes the effect that certain cost drivers and environmental factors have on productivity.

The solution to early estimation proposed in this paper is most applicable to large organisations for a number of reasons. Firstly larger organisations are more likely to have enough 'similar' projects to provide an adequate case-base. Secondly, local calibration is necessary for consistent quantification and to minimise subjectivity. Lastly, human factors have been identified as very important to successful delivery of a software development project. Larger organisations may have formalised procedures for personnel and project evaluation which can provide quantified inputs to the case representation.

Extending the use of CBR for cost estimation to operate across broader domains involves similar problems of representation. There is a need to identify abstract features that capture similarity across domains. It may be prudent to pursue interactive adaptation in reusing cases from different domains.

6. References

ALBRECHT A. & GAFFNEY J. 1983 "Software function, source line of code and development effort prediction : a software science validation" IEEE Transactions on Software Engineering 9(6) p.639-648

BARKI H., S. RIVARD & J. TALBOT 1993 "Towards an assessment of software development risk" Journal of Management Information Systems 10 (2) p. 203-225

BARLETTA R. 1991 "An introduction to case-based reasoning" AI Expert, 6(8) pp42-49.

BISIO R. & F. MALABOCCHIA 1995 Cost estimation of software projects through case-base reasoning" Case-Based Reasoning Research and Development. First International Conference, ICCBR-95 Proceedings p11-22.

BOEHM B 1981 Software Engineering Economics, Prentice Hall

BOEHM B. & P. PAPACCIO 1988 "Understanding and controlling software costs" IEEE Transactions on Software Engineering 14 (10) p.1462-1477

BREDERO R., CARACOGLIA G., JAGGERS C., KOK P., TATE G. & VERNER J. 1989 "Comparative evaluation of existing cost estimation tools" Mermaid report D7.1Y

CHARZOGLOU P. & L. MACAULEY 1996 " A review of existing models for project planning and estimation and the need for a new approach" International Journal of Project Management 14(3) p.173-183

CUELENAERE A., M. VAN GENUCHTEN & F. HEEMSTRA 1987 "Calibrating a software cost estimation model: why an how" Information and Software Technology 29(10) p. 558-567.

FINNIE G.R., & WITTIG G.E., 1996, "AI Tools for Software Development Effort Estimation" in Proceedings of the Conference on Software Engineering: Education and Practice, University of Otago, 113-120.

FINNIE G.R., WITTIG G.E. & DESHARNAIS J-M 1997 "Estimating software development effort with case-based reasoning", Proceedings of International Conference on Case-Based Reasoning, D. Leake, E. Plaza, (Eds) p. 13-22

HEEMSTA F. 1992 "Software cost estimation" Information and Software Technology 34 (10) p. 627-639

HENNESSY D. & D. HINKLE 1992 "Applying case-based reasoning to autoclave loading" IEEE Expert, 7 (5) pp.21-26.

JACK R. & MANNION M. 1995 "Improving the software cost estimation process" Software Quality Management III Vol 1 p.245-256

JEFFREY D. 1987 "A software development productivity model for MIS environments" Journal of Systems and Software 7 p.115-125

KEMERER C. 1987 "An empirical validation of software cost models" CACM 30 (5)

KENDALL R. & E. LAMB 1977 "Management perspective on programs, programming and productivity" presented at Guide 45, Atlanta, GA p.201-211

KOLODNER J. 1993 Case-based reasoning, Morgan Kaufmann, California

KUSTERS R., M. VAN GENUCHTEM & F. HEEMSTRA 1990 "Are software cost-estimation models accurate?" Information and Software Technology 32 (3) p.187-190.

LEDERER A. & J PRASAD 1992 "Nine management guidelines for better cost estimating" CACM 35 (2) p. 51-59

MUKHOPADHYAY T., S. VICINANZA & M. PRIETULA 1992 "Examining the feasability of a case-based reasoning model for software effort estimation" MIS Quarterly 16(2) p.155-171

NOTH T. & M KRETZSCHMAR 1984 "Estimation of software development projects" Springer Verlag (in German)

PRIETULA M., S. VICINANZA & T. MUKHOPADHYAY 1996 "Software effort estimation with a case-based reasoner" Journal of Experimental and Theoretical Artificial Intelligence 8(3-4) p.341-63

SHEPPERD M. & C. SCHOFIELD 1996 "Effort estimation by analogy: a case study" Presented at the European Control and Metrics Conference, Wilmslow, UK, May 1996. [Online] Available: http://xanadu.bournemouth.ac.uk/ComputingResearch/ChrisSchofield/angel/papers/ESCOM96.html

SHEPPERD M., C. SCHOFIELD & B. KITCHENHAM 1996 "Effort estimation using analogy" Proceedings of the 18th International Conference on Software Engineering p. 170-178

SUBRAMANIAN G. & S. BRESLAWSKI 1994 "The importance of cost drivers used in cost estimation models: Perceptions of project managers" Proceedings of 1994 Information Resources Management Association International Conference, San Antonio TX, USA p503-507

SUBRAMANIAN G. & S. BRESLAWSKI 1995 "An empirical analysis of software effort estimate alterations" Journal of Systems and Software 31 (2) p135-141

VICINANZA S., T. MUKHOPADHYAY & M. PRIETULA 1991 "Software effort estimation: an exploratory study of expert performance" Information Systems Research 2(4) p.243-262

WATSON I. & F. MARIR 1994 "Case-based reasoning: a review", The Knowledge Engineering Review 9(4) p. 327-354.

WRIGLEY C.D. & A.S. DEXTER 1987 "Software development estimation models: A review and critique" Proceedings of the ASAC Conference, University of Toronto, p. 125-138